## REMARKS:

Claims 1-55 were presented for examination and were pending in this application. In an Official Action dated April 13th, 2006, claims 1-55 were rejected. In an amendment and reply dated June 13th, 2006, Applicants requested amendments to claims 17 and 19 and withdrawal of all outstanding rejections. In an Advisory Action mailed on June 29th, 2006, the requested amendments to claims 17 and 19 were not entered and the rejections not withdrawn. Applicants thank Examiner for examination of the claims pending in this application and address Examiner's comments below.

Applicants herein amend claims 17 and 19. These changes are believed not to introduce new matter, and their entry is respectfully requested. The claims have been amended to expedite the prosecution of the application in a manner consistent with the Patent Office Business Goals, 65 Fed. Reg. 54603 (Sept. 8, 2000). In making these amendments, Applicants have not and do not narrow the scope of the protection to which Applicants consider the claimed invention to be entitled and do not concede that the subject matter of such claims was in fact disclosed or taught by the cited prior art. Rather, Applicants reserve the right to pursue such protection at a later point in time and merely seek to pursue protection for the subject matter presented in this submission.

Based on the above Amendment and the following Remarks, Applicants respectfully request that Examiner reconsider all outstanding rejections, and withdraw them.

## Response to Rejection Under 35 USC 103

In the 4th paragraph of the Office Action, claims 1, 29-32, and 42-46 were rejected under 35 USC § 103 as allegedly being obvious over U.S. Patent No. 6,542,991 to Joy el at ("Joy") in view of U.S. Patent No. 6,317,774 to Jones et al ("Jones"). This rejection is respectfully traversed.

Independent claim 1 recites:

> a hardware thread scheduler for identifying which of said program threads said processor executes and configurable to allocate available processing time of the processor among at least the first and second program threads by causing thread-switching at a fixed time according to a predetermined fixed schedule, said schedule specifying that the first thread should be allocated processing time every first number of cycles and that the second thread should be allocated processing time every second number of cycles, wherein said first number of cycles is not equal to said second number of cycles.

Similarly, independent claim 46 recites:

> "A computer based method for switching between program contexts in a multithreading pipelined processor having a hardware thread selector and an execution pipeline, the method comprising:...
> switching the processor from the first thread state to the second thread state by coupling the execution pipeline from the first set of data storage devices to the second set of storage devices via the hardware thread selector at a fixed time according to a predetermined fixed execution schedule, said execution schedule specifying that the processor should switch to the first thread state every first number of cycles and that the processor should switch to the second thread state every second number of cycles, wherein said first number of cycles is not equal to said second number of cycles."

The hardware thread scheduler is configurable to cause the processor to switch from one thread to another thread at a fixed time according to a predetermined fixed schedule. This is greatly beneficial because it provides a predictable execution time for threads.

As correctly noted by the Examiner, Joy does not teach thread-switching at a fixed time according to a predetermined fixed schedule. Instead, Joy discloses "oblivious thread switching", in which the thread executed by the processor is switched every N cycles. (Col. 17, ln. 1-4) In the oblivious thread switching disclosed by Joy, a first thread would be executed every K cycles (where K is N times the number of threads in the oblivious switching rotation), a second thread would be executed every K cycles, and so on.

The deficiencies of Joy are not rectified by the disclosure of Jones. Jones merely describes a multithreading operating system scheduling feature using a directed acyclic graph of nodes (col. 2, lines 51-54). Jones discloses a method for determining at a software level which application to execute when the processor becomes available (see Abstract). Applications can request reservations, which specify a number of time units for which the application will be executed. Applications can also specify reservation windows, which specify that the application will execute at least every number of time units (see col. 5, lines 8-17).

The combination of Joy and Jones, as suggested by the Examiner, is deficient at least for failing to disclose the hardware thread scheduler of the claim. Jones and Joy, both alone and in combination, fail to disclose a hardware thread scheduler configurable to allocate available processing time among a first and a second program thread according to a predetermined fixed schedule, said schedule specifying that the first thread should be allocated processing time every first number of cycles and that the second thread should be allocated processing time every second number of cycles, wherein said first number of cycles is not equal to said second number of cycles. The thread scheduler of Joy, as admitted by the

Examiner, is explicitly not configurable to allocate available processing time in the manner described in the claim. Jones, describing a software-based activity scheduler, does not disclose any hardware thread scheduler whatsoever, much less a hardware thread scheduler configurable as in the claim.

Jones also fails to disclose a schedule specifying that a first thread should be allocated processing time every first number of cycles and that a second thread should be allocated processing time every second number of cycles. The reservations of Jones are given in "arbitrary time units" (see col. 6, line 67), which roughly specify the percentage of total processor time and the general time frame during which an application will run (see col. 5, lines 35-37). Jones makes no suggestion that a time unit could correspond to a fixed number of cycles. In fact, a time unit in Jones *cannot* correspond to a fixed number of cycles, because, as a software program executing on a conventional CPU, the software scheduling unit of Jones cannot predict and does not control the number of cycles that will be lost in determining and performing a switch from one application to the next. As software, the application scheduler of Jones must itself be executed on the processor to determine activity switches. Jones teaches that the application scheduler executes for a "bounded amount of time" (see Col. 2, lines 30-34) during which the application scheduler consumes cycles. The imprecise resolution of software-based scheduling and the need to execute the software application scheduler itself render Jones incompatible with cycle-specific scheduling. A reservation indicating that an application should be executed every 100 time units, for example, could not indicate a specific number of cycles, because any varying number of cycles could be lost in the execution of the application scheduler and in performing the switch to once a determination has been made. The software-based

application scheduler of Jones assigns processing time and executes at too high a level to allocate specific numbers of instruction cycles.

To establish a prima facie case of obviousness, each and every element of the claimed invention must be taught or suggested by the references. However, the combination of Joy and Jones has been shown to be deficient for failing to disclose either "a predetermined fixed schedule specifying that the first thread should be allocated processing time every first number of cycles and that the second thread should be allocated processing time every second number of cycles, wherein said first number of cycles is not equal to said second number of cycles" or "a hardware thread scheduler...configurable to allocate available processing time of the processor among at least the first and second program threads by causing thread-switching at a fixed time" according to such a schedule.

Furthermore, the proposed modification of a reference cannot change the principle of operation of a reference (see MPEP 2143.01). The only *scheduled* thread switching taught by Joy is the oblivious thread switching, which switches threads every N cycles. In the oblivious thread switching of Joy, "individual flip-flops locally determine a thread switch without notification of stalling" (see col. 3, lines 33-35) and is the switching is "typically implemented using a simple counter for counting cycles between switches" (see col. 7, lines 4-5). Joy teaches oblivious thread switching determined by low-level hardware circuits. The flip-flop-implemented thread switch of Joy is principally incompatible with Jones, which requires execution of a software scheduler to determine application switches. The Examiner has cited no art explaining how the individual flip-flops of Joy could be modified, for example, to construct a scheduling graph, to evaluate and in some cases refuse reservation

requests, to traverse nodes, to add nodes to the scheduling graph, or to perform the various other tasks required to implement the activity scheduling of Jones. To implement a hardware thread scheduler capable of executing the software methodology of Jones would require a substantial reconstruction and redesign of the hardware shown in Joy, as well as changes in the basic principle under which Joy was designed to operate.

For at least these reasons, Applicants submit that the rejection of claims 1 and 46 under 35 USC 103 is improper and should be withdrawn.

As claims 29-32 and 42-45 are dependent from claim 1, all arguments presented with regard to claim 1 are hereby incorporated so as to apply to claims 29-32 and 42-45. For at least these reasons, Applicants submit that the rejection of claims 29-32 and 42-45 should be withdrawn.

In the 15th paragraph of the Office Action, claims 2-3, 13-17, 19, and 21-24 were rejected under 35 USC § 103 as allegedly being obvious over Joy in view of U.S. Patent No. 6,493,741 to Emer et al ("Emer"). This rejection is respectfully traversed in view of the amended claims.

Independent claim 17, as amended, recites:

> "A computer based system for switching between program contexts comprising:
>
> a pipelined processor capable of having a first program thread and a second program thread in an execution pipeline having a thread selection hardware, the execution pipeline including a set of stages for executing instructions and configured to execute a single instruction at each different stage of the set of stages;...
>
> a hardware thread scheduler for identifying which of said program threads said pipelined processor executes and configurable to allocate available processing time of the pipelined processor among at least the first and second program threads according to an execution schedule;

wherein said thread selection hardware in the pipelined processor switches from said first thread state to said second thread state between consecutive instruction cycles in response to the hardware thread scheduler identifying which of said program threads said pipelined processor executes."

Similarly, independent claim 19, as amended, recites:

"A computer based method for switching between program contexts in a multithreading pipelined processor having a hardware thread selector and an execution pipeline, the execution pipeline including a set of stages for executing instructions and configured to execute a single instruction at each different stage of the set of stages, the method comprising...:

switching the pipelined processor from executing the first program thread to executing the second program thread between the end of an execution cycle and before the beginning of a next consecutive execution cycle by coupling the execution pipeline from the first set of data storage devices to the second set of storage devices via the hardware thread selector."

An execution pipeline including a set of stages advantageously allows several instructions to be in the pipeline simultaneously, each at a different stage. Switching program threads in a pipelined processor within an instruction cycle and switching thread states between consecutive instruction cycles beneficially allows switching between one program context and another without incurring any time penalty. Switching from one thread to another in a pipelined processor within an execution cycle and before the beginning of a next consecutive execution cycle beneficially allows switching to occur without the loss of any execution cycles.

As correctly noted by the Examiner, Joy does not teach switching the pipelined processor from said first thread state to said second thread state between consecutive instruction cycles or switching the pipelined processor from executing the first program thread to executing the second program thread between the end of an execution cycle and

before the beginning of a next consecutive execution cycle. Joy teaches that "the thread switch logic generates the TID signal with a thread switch delay or overhead of one processor cycle." (See col. 16, ll. 1-5 and ll. 61-62.) By teaching a switch delay of one processor cycle, Joy teaches that it is not possible to switch *between consecutive instruction cycles* in response to the hardware thread scheduler.

The deficiencies of Joy are not rectified by Emer. The thread switching taught by Emer is directed towards "superscalar, multithreading, and simultaneous multithreading architectures." (See col. 1, ll. 33-35.) Emer does not disclose switching a pipelined processor capable of having a first program thread and a second program thread in an execution pipeline, the execution pipeline includes a set of stages for executing instructions, each instruction at a different stage of the set of stages, each stage of the set of stages executes a single instruction. As clearly illustrated in Figure 1, the architectures disclosed by Emer are configured to receive multiple instructions in a single processor cycle for the same stage, the issue stage. Therefore, at least the issue stage of Emer operates on multiple instructions in a single cycle. Emer does not disclose a pipelined processor that executes a single instruction in each stage of the pipeline as recited in the claims, and Emer certainly does not disclose switching such a pipelined processor between consecutive instruction cycles.

To establish a prima facie case of obviousness, each and every element of the claimed invention must be taught or suggested by the references. However, the combination of Joy and Emer has been shown to be deficient for failing to disclose a "pipelined processor capable of having a first program thread and a second program thread in an execution pipeline having a thread selection hardware, the execution pipeline including a set of stages

for executing instructions and configured to execute a single instruction at each different stage of the set of stages ... and thread selection hardware in the pipelined processor switches from said first thread state to said second thread state between consecutive instruction cycles" and for failing to disclose a "pipelined processor having ...an execution pipeline, the execution pipeline including a set of stages for executing instructions and configured to execute a single instruction at each different stage of the set of stages ... switching the pipelined processor from executing the first program thread to executing the second program thread an execution cycle and before the beginning of a next consecutive execution cycle".

Furthermore, the proposed modification of a reference cannot render the proposed combination inoperable for its intended purpose (see MPEP 2143.01). Emer and Joy are directed towards entirely different hardware architectures. It is not apparent how one of ordinary skill in the art would combine the single-processor vertically-threaded processor of Joy with the superscalar, multithreading, and simultaneous multithreading architectures of Emer. The single-processor vertically-threaded processor of Joy has a single pipeline shared among a plurality of threads. The thread switch logic of Joy selects at most one thread at any given time as the active thread, and the "currently active thread...supplies data to functional blocks connected to the pipeline." (See Joy Col. 8, ll. 19-25.) The architectures illustrated in Emer are capable of receiving multiple instructions in a single clock cycle. Therefore, the thread switch logic of Joy, which merely activates one thread for execution on the functional blocks of the pipeline, would not be able to switch the superscalar, multithreading, and simultaneous multithreading processors of Emer.

Moreover, Joy *explicitly* states that an overhead delay of one or more clock cycles is necessary for the single-processor vertically-threaded processor. (See col. 16, ll. 1-5 and ll. 61-62.) Joy teaches that such a delay is necessary to switch the state of the processor. Therefore, the single-processor vertically-threaded processor of Joy is fundamentally incompatible with *any* switching that causes the processor to switch threads every processor cycle. Switching every cycle with a one cycle overhead would prohibit any forward execution progress, rendering the proposed combination inoperable for its intended purpose.

For at least these reasons, Applicants submit that the rejection of claims 17 and 19 under 35 USC 103 is overcome and should be withdrawn.

As claims 2-3 and 13-16 are dependent from claim 17, and claims 21-24 are dependent from claim 19, all arguments presented with regard to claims 17 and 19 are hereby incorporated so as to apply to claims 2-3, 13-16 and 21-24. For at least these reasons, Applicants submit that the rejection of claims 2-3, 21-24, and 13-16 should be withdrawn.

In the 28th, 30th, and 37th paragraphs of the Office Action, Examiner rejects claims 4-12, 18, 20, 25-28, under 35 USC § 103(a) as allegedly being unpatentable in view of Joy, Emer, and various combinations of U.S. Patent No. 6,567,839 to Borkenhagan et al. ("Borkenhagan"), U.S. Patent No. 6,085,215 to Ramakrishnan et al. ("Ramakrishnan"), U.S. Patent No. 6,026,503 to Gutgold et al. ("Gutgold").

Similarly, in the 41st, 43rd, and 54th paragraphs of the Office Action, Examiner rejects claims 33-41, and 47-55 under 35 USC § 103(a) as allegedly being unpatentable in view of

Joy, Jones, and various combinations of Borkenhagen, Ramakrishnan, and Gutgold. These rejections are respectfully traversed.

As claims 4-12 and 18 are dependent on claim 17, all arguments advanced above with respect to claim 17 are hereby incorporated so as to apply to claims 4-12 and 18. As claims 20, and 25-28 are dependent from claim 19, all arguments advanced above with respect to claim 19 are hereby incorporated so as to apply to claims 20, and 25-28.

Claim 17 and its dependent claims have been shown to be patentable over the combination of Joy and Emer, at least because the combination fails to disclose a "pipelined processor capable of having a first program thread and a second program thread in an execution pipeline having a thread selection hardware, the execution pipeline including a set of stages for executing instructions and configured to execute a single instruction at each different stage of the set of stages ... thread selection hardware in the pipelined processor switches from said first thread state to said second thread state between consecutive instruction cycles". Similarly, Claim 19 and its dependent claims have been shown to be patentable over the combination of Joy and Emer, at least because the combination fails to disclose "a pipelined processor having ...an execution pipeline, the execution pipeline including a set of stages for executing instructions and configured to execute a single instruction at each different stage of the set of stages, ... switching the pipelined processor from executing the first program thread to executing the second program thread an execution cycle and before the beginning of a next consecutive execution cycle". These deficiencies are not remedied by the various disclosures, either alone or in combination, of Borkenhagen, Ramkrishnan, and Gutgold.

As claims 33-41 are dependent on claim 1, all arguments advanced above with respect to claim 1 are hereby incorporated so as to apply to claims 33-41. As claims 47-55 are dependent from claim 46, all arguments advanced above with respect to claim 46 are hereby incorporated so as to apply to claims 47-55.

Claim 1 and its dependent claims have been shown to be patentable over Joy and Jones, at least because the combination fails to disclose "a hardware thread scheduler for identifying which of said program threads said processor executes and configurable to allocate available processing time of the processor among at least the first and second program threads by causing thread-switching at a fixed time according to a predetermined fixed schedule, said schedule specifying that the first thread should be allocated processing time every first number of cycles and that the second thread should be allocated processing time every second number of cycles, wherein said first number of cycles is not equal to said second number of cycles." Similarly, Claim 46 and its dependent claims have been shown to be patentable over Joy and Jones, at least because the combination fails to disclose "switching the processor from the first thread state to the second thread state by coupling the execution pipeline from the first set of data storage devices to the second set of storage devices via the hardware thread selector at a fixed time according to a predetermined fixed execution schedule, said execution schedule specifying that the processor should switch to the first thread state every first number of cycles and that the processor should switch to the second thread state every second number of cycles, wherein said first number of cycles is not equal to said second number of cycles." These deficiencies are not remedied by the various disclosures, either alone or in combination, of Borkenhagen, Ramkrishnan, and Gutgold.

Examiner has cited Borkenhagen as allegedly teaching a processor switching between states by changing a state selection register. Borkenhagen discloses switching threads in response to a cache miss or external interrupt signal (Col. 6, ln. 22-42), said switches incurring the conventional "latency and performance penalties associated with switching threads." (Borkenhagen, col. 15, ln . 37-48). Borkenhagan does not disclose the deficiencies of Joy or the deficiencies of Joy in combination with Emer cited above.

Likewise, Ramakrishnan is cited to make up for Joy's lack of "thread identifier for identifying at least one hard-real-time (HRT) thread and at least one non-real-time thread" limitation. Ramakrishnan simply describes a software scheduler that uses a round robin approach to thread scheduling using conventional context switching. See Ramakrishnan, col. 9, ln. 9-10. The switching in Ramakrishnan, like in Borkenhagen, is conventional context switching that involves "an associated overhead in invoking the new thread." (Ramakrishnan, col. 12, lines 61-62, see also, col. 13, lines 6-7 "avoid time consuming context switching"). Ramakrishnan does not disclose the deficiencies of Joy or the deficiencies of Joy in combination with Emer cited above.

Finally, Gutgold is relied upon to provide the different access speed memory devices recited in claims 10-12 and 39-41 that are not explicitly described in Joy, Ramakrishnan, or Emer. However, the combined reference still fails to teach or suggest the fixed time or consecutive-cycle context switching recited in the claims. Gutgold does not describe any context switching. Aside from the fact that Gutgold describes a microprocessor controlled system and associated microprocessor system components, Applicants see no other relation to Applicants' invention.

Accordingly, for at least the reasons set forth above, Applicants submit that claims 4-12, 18, 20, 25-28, 33-41, and 47-55 are patentable over the cited combined references and request that these rejections be withdrawn.

## Conclusion

In sum, Applicants respectfully submit that claims 1 through 55, as presented herein, are patentably distinguishable over the cited references. Therefore, Applicants request reconsideration of the basis for the rejections to these claims and request allowance of them.

In addition, Applicants respectfully invite Examiner to contact Applicants' representative at the number provided below if Examiner believes it will help expedite furtherance of this application.

Respectfully Submitted,
NICHOLAS J. KELSEY, ET AL.

Date: 8/14/2006 By: _____

Hector J. Ribera, Attorney of Record
Registration No. 54,397
FENWICK & WEST LLP
801 California Street
Mountain View, CA 94041
Phone: (650) 335-7192
Fax: (650) 938-5200
E-Mail: hribera@fenwick.com